Documentación de los programas, tarea 1 elo329.

Pascal Sigel- Javier Cabezas

abril-2014

1 Introducción:

El proyecto es un simulador de colisiones de bolas las cuales pueden interactuar con resortes y osciladores, consta de una interfaz gráfica y audio para las colisiones. Además se grafica la energía cinética, energía potencial y la energía mecánica del sistema.

Cómo resultado se pueden tener dos opciones, un .class ejecutable llamado PhysicsLab.class ó un .class para ser usado en Applet llamado PhysicsLabApplet.class

Para más detalles leer el archivo readme.txt.

En la próxima sección se explicarán las clases más importantes que componen al proyecto.

1.1 Clases e interfaces:

1.1.1 Clase PhysicsLab/PhysicsLabApplet:

La clase PhysicsLab es la clase main del proyecto, esta clase manipula el objeto de tipo MyWorld creándolo y dándole los elementos que interaccionarán en este "mundo" simulado.

PhysiscsLabApplet hace lo mismo pero para la applet.

1.1.2 Clase MyWorld/MyWorldApplet:

Esta clase es la que maneja la interacción entre los elementos del mundo simulado, el método principal de esta clase es el método **ActionPerformed** el cual a través de un Timer es activado y hace transcurrir la simulación.

1.1.3 Clase MyWorldView:

Este objeto se encarga que todos los elementos contenidos en el mundo sean dibujados en el panel.

1.1.4 Clase GraphicPane:

Se encarga de generar los gráficos en tiempo real.

1.1.5 Clase Collide_sound/Collide_sound_Applet:

Este objeto es el encargado de generar el audio en las colisiones de las bolas.

1.1.6 PhysicsElement:

Los elementos físicos son una clase abstracta de la cual nacen los elementos como bolas, resores, bloques y cualquier otro elemento físico.

1.1.7 Ball:

Esta clase se extiende (desciende) de la clase PhysicsElement y representa una bola la cual tiene masa, velocidad y posicion. En esta clase se calcula el próximo estado.

1.1.8 Spring:

Esta clase se extiende (desciende) de la clase PhysicsElement y representa un resorte el cual efectúa fuerzas de hook.

1.1.9 Oscillator:

Esta clase se extiende (desciende) de la clase PhysicsElement y representa un oscilador, que tiene definida su posición en el tiempo según una amplitud y una frecuencia dada.

1.2 BallView/SpringView/OscillatorView

Las clases con extensión View se encargan de dibujar en el panel a sus respectivos objetos.

1.3 Experimento y resultados:

Lease el archivo README para ejecutar el programa.

Bajo una ejecución normal el resultado del programa normal puede ser parecido a la imagen siguiente:

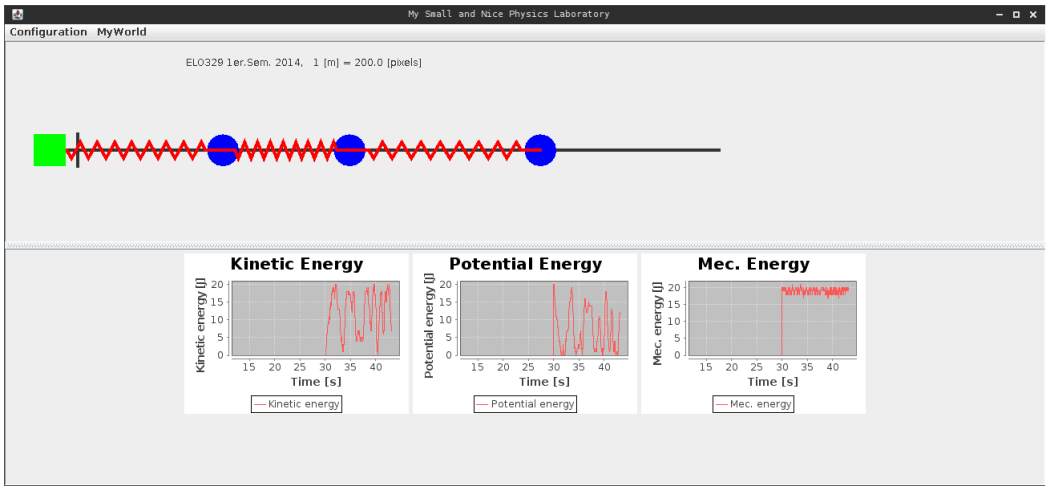

Figure 1: Simulación de prueba.

También puede ser ejecutado a través de appletview y se tendría un resultado como el de la siguiente figura:

1.4 Problemas que ocurrieron:

1.4.1 Problema con audio:

Los applet no aceptan la reproducción de audio que no son hechos a través de su propia librería, esto implicó mucho gasto de tiempo y energías hasta que se replicó la idea que se presenta en tic tac toe (http://profesores.elo.utfsm.cl/ agv/elo

1.4.2 Problema con Jfreechart y su inclusión en el proyecto:

Para el uso de Jfreechart se deben descargar los .jar que desde su sitio web se pueden descargar, el problema es que para hacerlos funcionar es necesario incluir los jar en la compilación, lamentablemente lo normal es que los programadores de java usen eclipse y la información de cómo incliur Jfreechart estaban explicadas para este IDE. La solución fue brindada por el profesor y corresponde al uso de la etiqueta -cp (classpath) para la compilación y ejecución.

1.4.3 Tamaño de Jfreechart:

Los .jar que son brindados por jfreechart tienen un tamaño de 1.5 mb y contienen una cantidad muy grande de clases incluidas, para reducir el tamaño del archivo se puede descomprimir el jar con “jar xf archivo.jar” y luego incluir los .class necesarios y suficientes. Se comenzó a hacer esto pero el tiempo necesario para encontrar todas las clases que son usadas es muy largo. Se perdió al rededor de 1 hora realizando esta labor y se dejó de hacer por lo que se incluyeron los .jar completos.

.